COSC 4P82 Assignment 1

Brett Terpstra
bt19ex@brocku.ca - 692021

February 16, 2024

Contents

1 Introduction

2 Symbolic regression

2.1 Introduction

2.2 Parameter Table

Parameter	Value
Runs	10
Population Size	5000
Generations	50
Training Set	N/A
Testing Set	N/A
Crossover Operator	Subtree Crossover
Mutation Operator	Grow Tree, Max Depth 4
Crossover Rate	0.9 or 1.0*
Mutation Rate	0.1 or 1.0*
Elitism	Best 2 or 0 individuals Survive*
Selection	Fitness Proportionate
Function Set	*, /, +, -, exp, log, sin, cos
Terminal Set	X, Ephemeral Value
Tree Initialization	Half and Half, Max Depth 2-6
Max Tree Depth	17
Raw Fitness	See Fitness Evaluation
Standardized Fitness	= Raw Fitness

*4 Tests were run, 0.9 crossover, 0.9 mutation with 0 elitism and 2 elitism, and 1.0 crossover, 1.0 mutation with 0 elitism and 2 elitism.

2.3 Fitness Evaluation

Fitness is evaluated by taking the absolute value of the predicted y value minus the actual y value. If the difference is less than a user provided (default 1.e15) value cutoff it is added to the fitness value. If the difference value is less than the float epsilon value ($\doteq 0$) the number of hits is incremented. Lower fitness values are preferred.

2.4 Fitness Plots

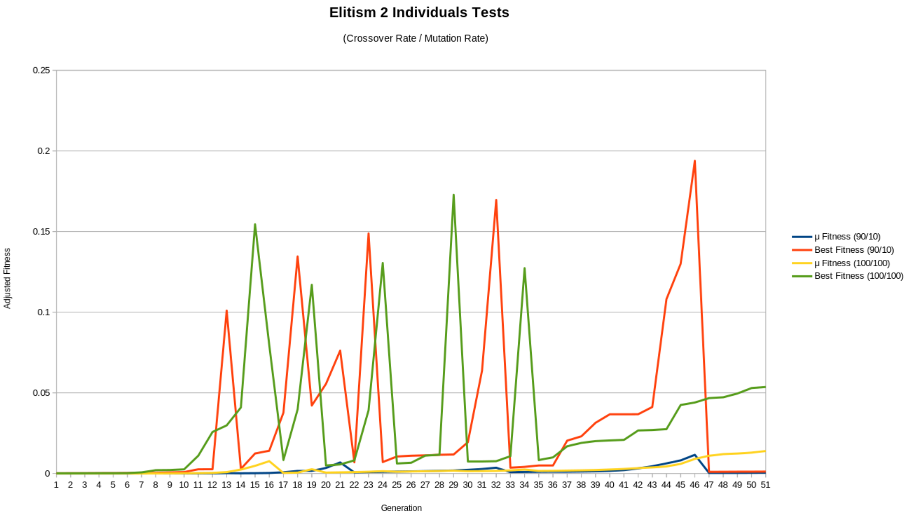


Figure 1: 2 Elites, 10 Runs Averaged

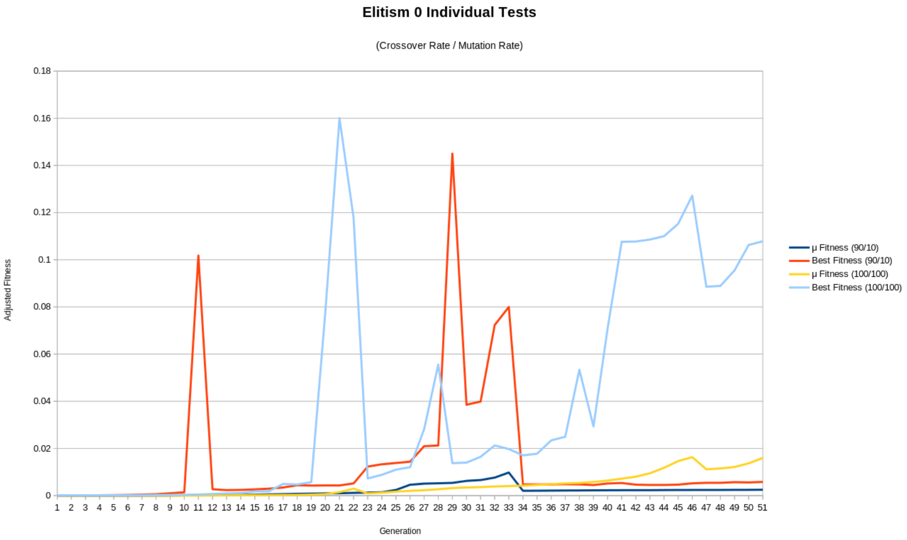


Figure 2: 0 Elites, 10 Runs Averaged

2.5 Analysis and Conclusion

The best average fitness of all the tests was 0.19384 using 0.9 crossover and 0.1 mutation.

3 Rice Classification

3.1 Introduction

3.2 Parameter Table

Parameter	Value
Runs	10
Population Size	5000
Generations	51
Training Set	Rice Classification (Cammeo and Osmancik)
Testing Set	Rice Classification (Cammeo and Osmancik)
Crossover Operator	Subtree Crossover
Mutation Operator	Grow Tree, Max Depth 4
Crossover Rate	0.9 or 0.9*
Mutation Rate	0.1 or 0.9*
Elitism	Best 2 individuals Survive
Selection	Fitness Proportionate
Function Set	*, /, +, -, exp, log
Terminal Set	area, perimeter, major, minor, eccentricity, convex, extent, Ephemeral Value
Tree Initialization	Half and Half, Max Depth 2-6
Max Tree Depth	17
Raw Fitness	See Fitness Evaluation
Standardized Fitness	= Raw Fitness

3.3 Fitness Evaluation

Tested on the input terminal values the GP produces a positive or negative value which is interpreted as either Cammeo (+) or Osmancik (-). Raw fitness is equal to the number of hits which is the number of correct identifications. The adjusted fitness is then calculated and subtracted from 1 in order to invert and produce the required lowest fitness better.

3.4 Fitness Plots

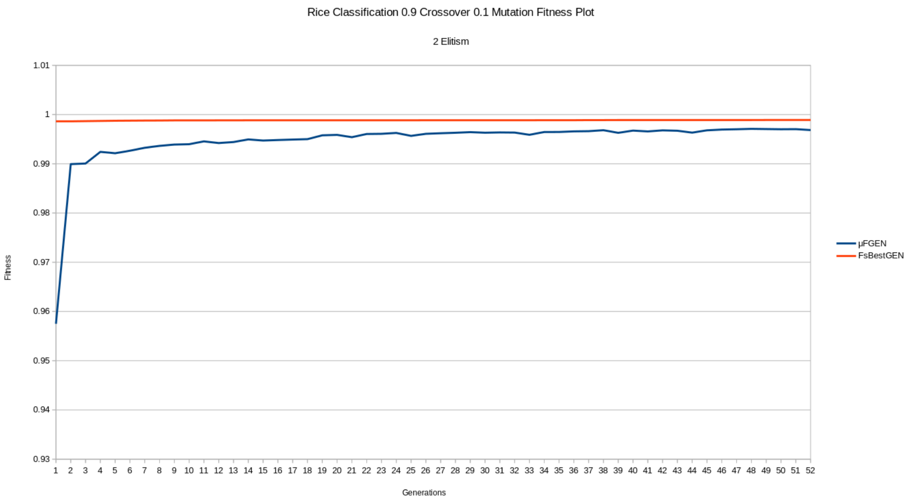


Figure 3: 2 Elites, 10 Runs Averaged

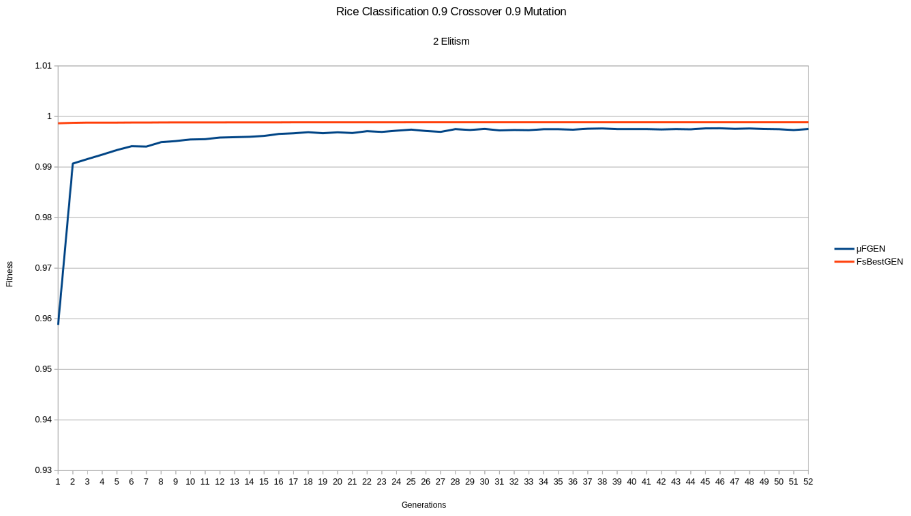
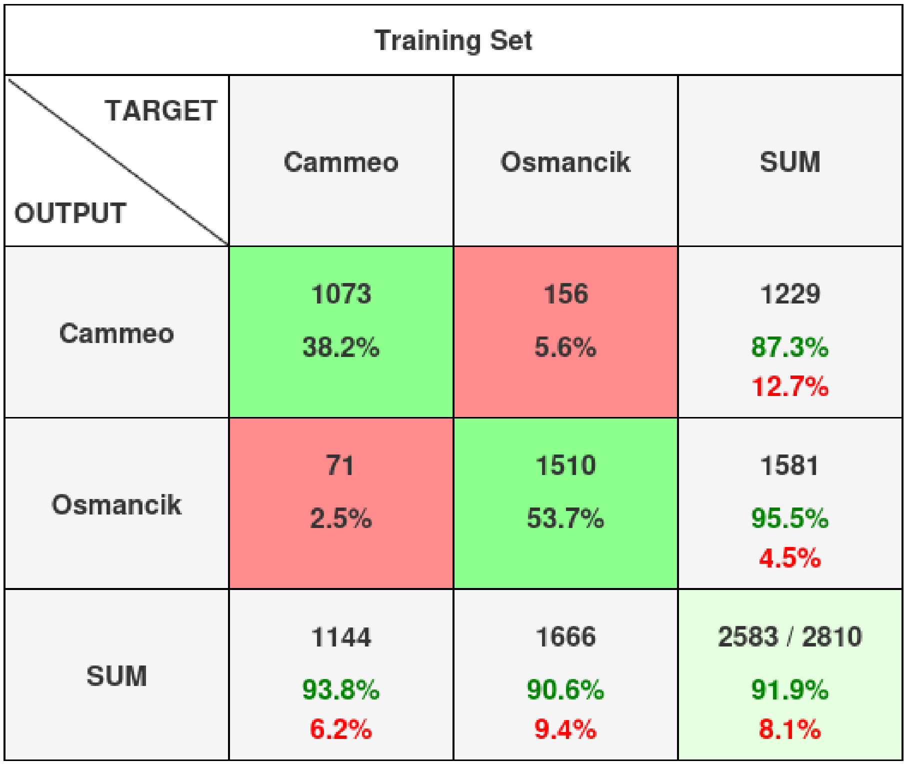


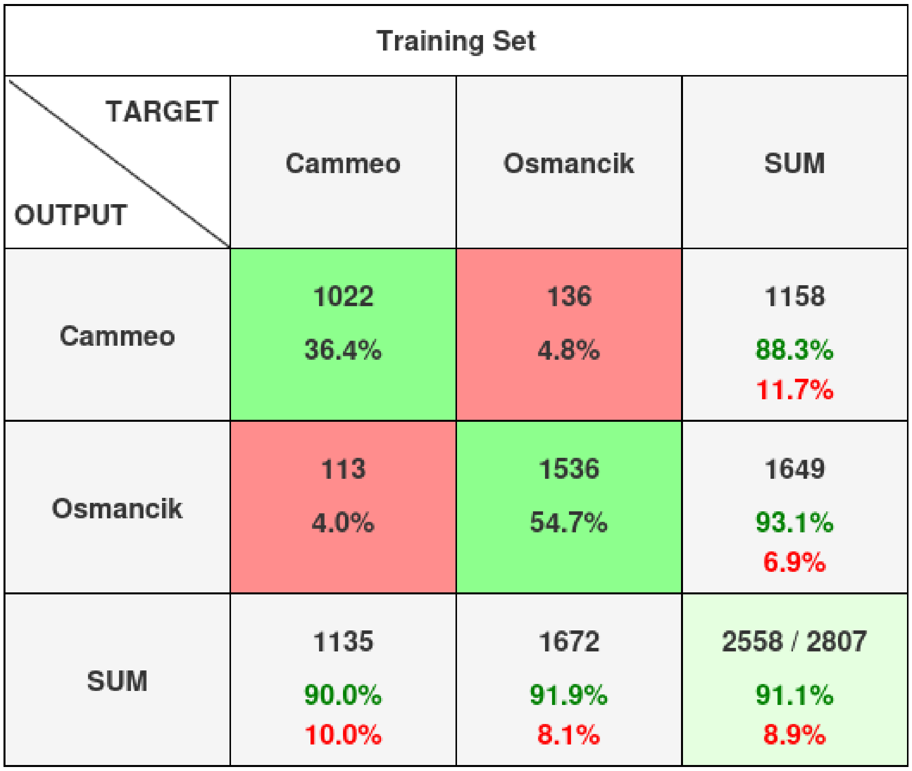
Figure 4: 2 Elites, 10 Runs Averaged

3.5 Confusion Matrix



Training Set			
TARGET OUTPUT	Cammeo	Osmancik	SUM
Cammeo	1073 38.2%	156 5.6%	1229 87.3% 12.7%
Osmancik	71 2.5%	1510 53.7%	1581 95.5% 4.5%
SUM	1144 93.8% 6.2%	1666 90.6% 9.4%	2583 / 2810 91.9% 8.1%

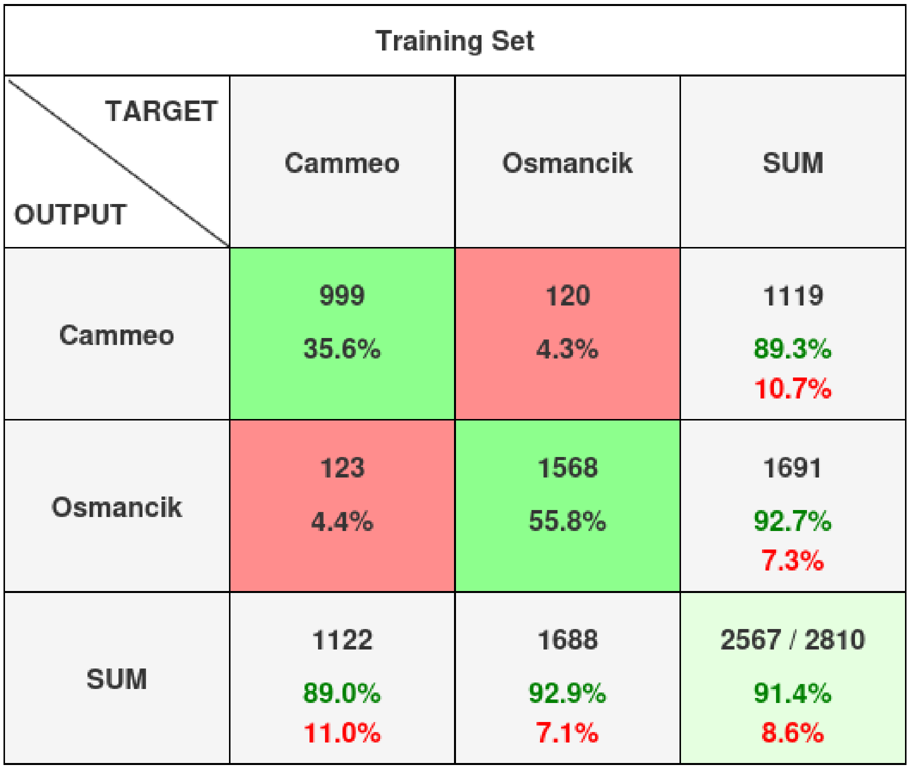
Figure 5: 0.9 Crossover 0.1 Mutation 2 Elites Best Program Results



Training Set

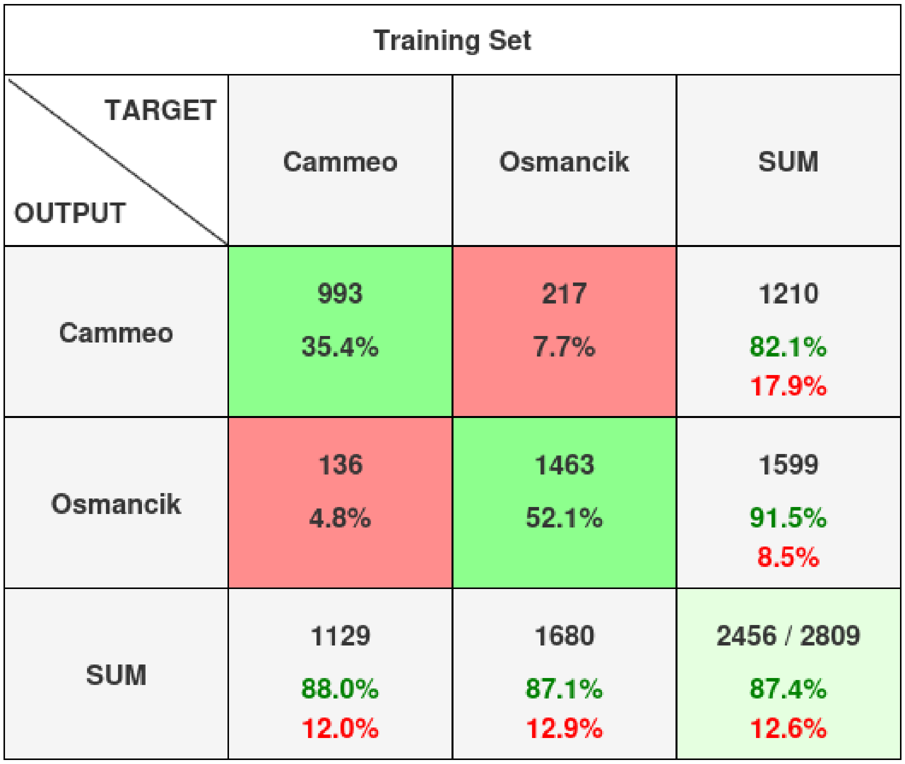
TARGET OUTPUT	Cammeo	Osmancik	SUM
Cammeo	1022 36.4%	136 4.8%	1158 88.3% 11.7%
Osmancik	113 4.0%	1536 54.7%	1649 93.1% 6.9%
SUM	1135 90.0% 10.0%	1672 91.9% 8.1%	2558 / 2807 91.1% 8.9%

Figure 6: 0.9 Crossover 0.1 Mutation 2 Elites 10 Run Average Results



Training Set			
TARGET OUTPUT	Cammeo	Osmancik	SUM
Cammeo	999 35.6%	120 4.3%	1119 89.3% 10.7%
Osmancik	123 4.4%	1568 55.8%	1691 92.7% 7.3%
SUM	1122 89.0% 11.0%	1688 92.9% 7.1%	2567 / 2810 91.4% 8.6%

Figure 7: 0.9 Crossover 0.9 Mutation 2 Elites Best Program Results



Training Set

TARGET OUTPUT	Cammeo	Osmancik	SUM
Cammeo	993 35.4%	217 7.7%	1210 82.1% 17.9%
Osmancik	136 4.8%	1463 52.1%	1599 91.5% 8.5%
SUM	1129 88.0% 12.0%	1680 87.1% 12.9%	2456 / 2809 87.4% 12.6%

Figure 8: 0.9 Crossover 0.9 Mutation 2 Elites 10 Run Average Results

3.6 Analysis and Conclusion

The best results found was a correct classification rate of 91.9%. On average the 0.9 crossover with 0.1 mutation produced the best results with the 0.9/0.9 best result almost being equal.

4 Compiling / Executing

This assignment was made for linux using GCC 13.2.0, however any C++17 compliant compiler should work. The minimum GCC version appears to be 8.5, meaning this assignment can be built on sandcastle.

```
cd your_path_to_this_source/
mkdir build
cd build
cmake ../
make -j 32
```

The actual assignment executable is called `Assignment_1` while the automatic run system is called `Assignment_1_RUNNER`. `Assignment_1_RUNNER` has a help menu with options but the defaults will work assuming you run from the build directory and are using part b only. If you want to build for Part A run `cmake -DPART_B=OFF` and run `Assignment_1_RUNNER` with `-b`

5 Conclusion

I made a few changes to lilgp, mostly memory fixes along with elitism with a number of individuals instead of a proportion. There appear to be some kind of issue in the GP, of which won't matter as assignment two will likely use my own gp system. I might look into it, but I was not aware there was an issue until compiling the stats here. My results have been generally positive, however, I did notice in the course of collecting data that at some point the Part A results stopped being consistently good however part B results have remained unchanged. Might have happened when I changed my custom random number seeder to not produce div by zero errors during testing. Could be anything. I don't like writing reports and have procrastinated on writing and instead have spent the last couple of weeks messing around with the GP. Fun fact a bunch of additions to my standard lib were made for this assignment. Next time will be better hopefully

6 References

Next assignment these will be proper. Latex is being annoying to setup for bib.

https://archive.ics.uci.edu/dataset/545/rice+cammeo+and+osmancik
http://garage.cse.msu.edu/software/lil-gp/